VI. CLAIMS

What is claimed is:

1. A method for emulating an instruction that switches the contents of a top stack register and a selected stack register, comprising:

overriding the dependency of the top stack register and the selected stack register;

executing a first instruction that moves the contents of the top stack register into the selected stack register in parallel with a second instruction that moves the contents of the selected stack register into the top stack register.

2. The method of claim 1 further comprising:

determining whether the top register is empty if a stack underflow exception occurs on a first attempt to execute the first and second instructions, and

if the top register is empty, replacing its contents with a QNaN, and

if the top register is not empty, replacing the contents of the selected register with a QNaN.

3. The method of claim 2 further comprising:

replacing the contents of the selected register with a QNaN if the top register is empty on the first attempt to execute the first and second instructions and if a stack underflow exception occurs on a second attempt to execute the first and second instructions.

4. A system comprising:

main memory storing an instruction set; and

a processor operably connected to main memory by a bus network, wherein the processor comprises:

a floating point unit;

a register stack;

dependency checking logic for determining whether instructions are executed sequentially or in parallel;

two execution units for executing instructions; and

ROM storing a micro-code handler that is invoked when two move instructions operating on the register stack are executed in parallel and cause a stack underflow exception.

5. In a processor based computer system having dependency checking logic and a register stack, a method comprising:

overriding the dependency logic such that move instructions associated with the stack registers may be executed in parallel;

executing the move instructions in parallel;

determining whether a stack underflow exception has occurred and if it has;

flushing the move instructions; and

invoking a micro-code handler algorithm that operates to allow execution of the move instructions in parallel without a stack underflow exception.

6. The method of claim 5, wherein the register stack comprises a top register and a selectable register, and wherein the act of invoking the micro-code handler further comprises:

determining whether the top register of the stack is empty and if it is, replacing its contents with an appropriate architectural response.

7. The method of claim 6, wherein the act of determining whether the top register of the stack is empty, further comprises:

replacing the contents of the selectable register with an appropriate architectural response if the top register of the stack is not empty.

8. The method of claim 7, further comprising:

executing the move instructions in parallel if the top or selectable register contents have be replaced with an appropriate architectural response.

9. The method of claim 7, further comprising:

replacing the contents of the selectable register with an appropriate architectural response if the contents of the top register have been replaced with an appropriate architectural response and a stack underflow exception has occurred.

10. A method for emulating an FXCH instruction, comprising:

providing a processor with a move ST(0) instruction and a move ST(i) instruction, wherein ST(0) denotes the top register of a stack and ST(i) denotes the *i*th register of the stack;

overriding the sequential dependency of the ST(0) and ST(i) registers;

executing the move ST(0) and move ST(i) instructions in parallel.

11. The method of claim 10, further comprising:

providing the move ST(0) instruction and move ST(i) instruction to respective execution units such that the instructions are executed substantially at the same time.

12. The method of claim 10, further comprising:

flushing the move ST(0) instruction and the move ST(i) instruction if a stack underflow exception occurs when the instructions are executed in parallel; and

invoking an algorithm that determines whether the stack underflow exception occurred because the ST(0) register, ST(i) register, or both were empty.

13. The method of claim 12, wherein the algorithm replaces an empty stack register with an appropriate architectural response.

14. The method of claim 13, wherein the algorithm replaces the ST(i) register with an appropriate architectural response if a stack underflow exception occurs the first time the move instructions are executed and the ST(0) register is not empty.

15. The method of claim 14, wherein the algorithm replaces the ST(i) register with an appropriate architectural response if a stack underflow exception occurs and the contents of the ST(0) register have been replaced with an appropriate architectural response.

16. A processor, comprising:

dependency checking logic; and

a register stack having a top register and a plurality of other registers;

wherein the processor is configured to:

override the dependency logic such that operations related to the register stack may be executed in parallel;

substantially simultaneously switch the contents of the top register of the register stack with the contents of another register of the register stack and vice versa, such that a hazard does not occur; and

execute an algorithm that replaces the contents of the top register, the other register, or both registers with an appropriate architectural response if an exception occurs when the contents of both registers are switched.

17. The processor of claim 16, wherein the appropriate architectural response is a QNaN.

18. The processor of claim 16 further comprising:

at least two execution units and a ROM.

19. The processor of claim 18, wherein the ROM stores a handler that is capable of implementing the appropriate architectural response.

20. A method of exchanging the contents of two registers, comprising:

overriding the dependency of the two registers; and

executing instructions that exchange the contents of the two registers in parallel and in lockstep.